Decentralized Diagnosis of Event-Driven Systems for Safely Reacting to Failures

Wenbin Qiu, Qin Wen and Ratnesh Kumar

Abstract—We introduce the notion of safe-codiagnosability, extending the notion of safe-diagnosability [8] to the decentralized setting. For a system, a certain sub-behavior is deemed safe (captured via a safety specification), and a further sub-behavior is deemed non-faulty (captured via a non-fault specification). Safe-codiagnosability requires that when the system executes a trace that is faulty, there exists at least one diagnoser that can detect this within bounded delay and also before the safety specification is violated. The above notion of safe-codiagnosability may also be viewed as an extension of the notion of codiagnosability [11], where the latter did not have any safety requirement. We show that safe-codiagnosability is equivalent to codiagnosability together with “zero-delay codiagnosability” of “boundary safe traces”. (A safe trace is a boundary safe trace if there exists a single-event extension that is unsafe.) We give an algorithm of polynomial complexity for verifying safe-codiagnosability. For a safe-codiagnosable system, the same methods as those proposed in [11] can be applied for off-line synthesis of individual diagnosers, as well as for on-line diagnosis using them.

Index Terms—Discrete-event systems, Fault diagnosis, Safety analysis, Decentralized systems

NOTE TO PRACTITIONERS

It is important that failures be diagnosed within a bounded delay of their occurrences and before any safety conditions are violated. The notion of safe diagnosability captures this requirement. The paper provides a condition a system must satisfy so that any failure can be safely diagnosed by one or more of the diagnosers acting independently of each other. Our results provide the insight that the safe diagnosability property can be separated into the property of diagnosability together with the normality of the “boundary safe behaviors”. The condition presented can be verified polynomially, improving the exponential complexity of an earlier work. The synthesis of a decentralized set of diagnosers as well as the on-line diagnosis using them is also of polynomial complexity.

I. INTRODUCTION

For discrete event systems (DESs), a certain model based approach for failure diagnosis is proposed in [14]. The property of diagnosability requires that once a failure has occurred, it be detected and diagnosed within bounded “delay” (within bounded number of transitions). The diagnosability can be tested polynomially as shown later in [3], [17]. Diagnosis in “state-based” setting was considered in [18], where the occurrence of a failure is modeled as reaching of certain “faulty states”. To facilitate generalization of failure specifications, linear-time temporal logic (LTL) based specification and diagnosis of its failure was proposed in [4]. A theory for failure diagnosis of repeatedly-occurring/intermittent failures was introduced in [6], and generalized to temporal logic setting in [5].

The above mentioned work dealt with centralized failure diagnosis. [2] considered the problem of distributed failure diagnosis based on a “coordinated decentralized architecture”, where local diagnosers do not communicate with each other directly, but send local information to a coordinator. Then the coordinator makes the final diagnosis.

Manuscript received ; revised . The research was supported in part by the National Science Foundation under the grants NSF-ECCS-0424048, NSF-ECCS-0601570, NSF-ECCS-0801763, and NSF-CCF-0811541.

W. Qiu is with the Comping Incorporated Sullivan Park Research Center, Painted Post, NY 14870 USA (e-mail: qiw@corn.com).

Q. Wen and R. Kumar are with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011 USA (e-mail: wenqin, rkumar@iastate.edu).

decision. In [11], we studied the problem of decentralized failure diagnosis, where the system failure is diagnosed by multiple local diagnosers. A notion of codiagnosability was introduced to capture the fact that the occurrence of any failure must be diagnosed within bounded delay by at least one local diagnoser using its own observations of the system execution. Polynomial algorithms were provided for (i) testing codiagnosability, (ii) computing the delay bound of diagnosis, (iii) off-line synthesis of diagnosers, and (iv) on-line diagnosis using them. An “inferencing based” approach to decentralized diagnosis is considered in [7], [16], and the notion of N-inference diagnosability has been introduced that is weaker than codiagnosability.

[15] considered a type of distributed diagnosis problem, where communication directly exists between local diagnosers, and is of unbounded delay. To address this problem [15] formulate the notion of decentralized diagnosability and proved its undecidability. Recent work has shown that distributed diagnosis under unbounded delay communication is decidable [12]. Methods for distributed diagnosis under bounded communication delay were proposed in [9], [10]. A distributed diagnosis problem with asymmetric communication is studied in [1], where communication is one-way and without delays.

In order to react to a failure in a timely fashion, while it is necessary that the failure be detected within a bounded delay, it is also needed that the detection occur before the system behavior becomes “unsafe”. To capture this additional requirement for failure detection, the notion of safe-diagnosability was introduced in [8]. We extend this notion to the decentralized setting, and introduce the notion of safe-codiagnosability.

For a system, a certain sub-behavior is deemed safe (captured via a safety specification), and a further sub-behavior is deemed non-faulty (captured via a non-fault specification). The safe behavior includes all of non-faulty behavior and some of post-fault behavior where system performance may be degraded but still tolerable. Safe-codiagnosability requires that when the system executes a trace that is faulty, there exist at least one diagnoser that can detect this within bounded delay and also before the safety specification is violated. The above notion of safe-codiagnosability may also be viewed as an extension of the notion of codiagnosability [11], where the latter did not have any safety requirement. We show that safe-codiagnosability is equivalent to codiagnosability together with “zero-delay codiagnosability” of “boundary safe traces”. (A safe trace is a boundary safe trace if there exists a single-event extension that is unsafe.) We give an algorithms of polynomial complexity for verifying safe-codiagnosability.

For a system, a certain sub-behavior is deemed safe (captured via a safety specification), and a further sub-behavior is deemed non-faulty (captured via a non-fault specification). The safe behavior includes all of non-faulty behavior and some of post-fault behavior where system performance may be degraded but still tolerable. Safe-codiagnosability requires that when the system executes a trace that is faulty, there exist at least one diagnoser that can detect this within bounded delay and also before the safety specification is violated. The above notion of safe-codiagnosability may also be viewed as an extension of the notion of codiagnosability [11], where the latter did not have any safety requirement. We show that safe-codiagnosability is equivalent to codiagnosability together with “zero-delay codiagnosability” of “boundary safe traces”. (A safe trace is a boundary safe trace if there exists a single-event extension that is unsafe.) We give an algorithms of polynomial complexity for verifying safe-codiagnosability. (The verification algorithm presented in [8] was based upon the structural property of a deterministic diagnoser, and had an exponential complexity owing to the exponential size of the diagnoser.) For a safe-codiagnosable system, the same methods as those proposed in [11] can be applied for off-line synthesis of individual diagnosers, as well as for on-line diagnosis using them.

II. NOTIONS AND PRELIMINARIES

Given an event set $\Sigma$, $\Sigma^*$ is used to denote the set of all finite length event sequences over $\Sigma$, including the zero length event sequence $\epsilon$. A member of $\Sigma^*$ is a trace and a subset of $\Sigma^*$ is a language. Given a language $K \subseteq \Sigma^*$, the complement of $K$, denoted $K^c \subseteq \Sigma^*$, is defined as $K^c := \Sigma^* - K$. If trace $t$ is a prefix of trace $s$, it is denoted as $t \leq s$. Given a language $K \subseteq \Sigma^*$, its prefix-closure, denoted $pr(K)$, is defined as, $pr(K) := \{ s \in \Sigma^* | \exists t \in K \text{ s.t. } s \leq t \}$, and $K$ is said to be prefix-closed if $K = pr(K)$. The supremal prefix-closed sublanguage of $K$, denoted $supP(K) \subseteq K$, is defined as, $supP(K) := \{ s \in K | pr(s) \subseteq K \}$. The quotient of $K_1$ with respect to $K_2$ is defined as $K_1/K_2 := \{ s \in \Sigma^* | \exists t \in K_2 \text{ s.t. } st \in K_1 \}$. The set of deadlocking traces of a language $K$ are those traces from...
which no further extensions exist in $K$, i.e., $s \in K$ is deadlocking trace if $\{s\}^{*} \cap K = \{s\}$.

A DES is modeled as a finite state machine (FSM) finite automaton (FA) $G$ and is denoted by $G(X, \Sigma, \alpha, x_0)$, where $X$ is the set of states, $\Sigma$ is the finite set of events, $x_0 \in X$ is the initial state, and $\alpha : X \times \Sigma \rightarrow 2^X$ is the transition function, where $\Sigma := \Sigma \cup \{\epsilon\}$. $G$ is said to be deterministic if $|\alpha(x, \cdot)| \leq 1$ and $|\alpha(x, \epsilon)| = 0$; otherwise, it is called nondeterministic. $(x, x', x)$ is a transition of $G$ if $x' \in \alpha(x, \cdot)$; it is an $e$-transition if $x = e$. Letting $e(x)$ denote the set of states reachable from $x$ in zero or more $e$-transitions, the transition function $\alpha$ can be extended from domain $X \times \Sigma$ to domain $X \times \Sigma^{*}$ recursively as follows: $\forall x \in X, s \in \Sigma^{*}, \exists i \in \Sigma, \alpha(x, \cdot) = e^{i}(x)$, and $\alpha(x, s \cdot) = e^{i}(\alpha(x, s))$. The generated language by $G$ is defined as $L(G) := \{s \in \Sigma^{*} | \alpha(x, s) \neq \emptyset\}$, i.e., it includes all traces that can be executed from the initial state of $G$. States reached by execution of deadlocking traces in $L(G)$ are called deadlocking states. A path in $G$ is a sequence of transitions $(x_1, \sigma_1, x_2, \sigma_2, \cdots, \sigma_{n-1}, x_n)$, where $\sigma_i \in \Sigma$ and $x_{i+1} \in \alpha(x_i, \sigma_i)$ for all $i \in \{1, \cdots, n-1\}$. The path is called a cycle if $x_1 = x_0$.

Given an automaton $G = \{X, \Sigma, \alpha, x_0\}$, the complete model of $G$ is defined as $G = \{X, \Sigma, \alpha, x_0\}$, where $X := X \cup \{F\}$, and $\Sigma$ is defined as follows.

$$\forall X \in \Sigma, \sigma \in \Sigma X, \sigma = \begin{cases} \alpha(X, \sigma) & \text{if } [X \in X] \land |\alpha(X, \sigma)| \neq 0 \\ F & \text{if } [F = F] \lor |\alpha(X, \sigma)| = 0 \end{cases}$$

Since all events are defined at each state, the complete model $G$ generates the language $\Sigma^{*}$, i.e., $L(G) = \Sigma^{*}$.

Given two automata $G = \{X, \Sigma, \alpha, x_0\}$ and $R = \{Y, \Sigma, \beta, y_0\}$, the synchronous composition of $G$ and $R$ is defined as, $G || R = \{X \times Y, \Sigma, \gamma, (x_0, y_0)\}$ such that $\forall (x, y) \in X \times Y, \sigma \in \Sigma$.

$$\gamma((x, y), \sigma) = \begin{cases} \alpha(x, \sigma) \times \beta(y, \sigma), & \text{if } \sigma \neq \epsilon; \\ \alpha(x, \epsilon) \times \{y\} \cup \{x \times \beta(\epsilon, \sigma)\}, & \text{otherwise}. \end{cases}$$

If the system execution is observed through a single global observer, we can define a global observation mask as $M : \Sigma \rightarrow \Sigma^{*}$ with $M(e) = \epsilon$, where $\Sigma := \Delta \cup \{\epsilon\}$ and $\Delta$ is the set of observed symbols. The definition of $M$ can be extended from events to event sequences inductively as follows.

$$M(e) = \epsilon, \forall \sigma \in \Sigma^{*}, \sigma \in \Sigma, M(\sigma \sigma) = M(\sigma) M(\sigma).$$

Given an automaton $G$ and mask $M$, $M(G)$ is the masked automaton of $G$ with each transition $(x, \sigma, x')$ of $G$ replaced by $(x, M(\sigma), x')$. The local observation masks associated with different local observers are defined as $M_i : \Sigma \rightarrow \Sigma_i$ ($i \in I = \{1, \cdots, m\}$), where $m$ is the number of local observers, $\Sigma_i := \Delta_i \cup \{\epsilon\}$ and $\Delta_i$ is the set of locally observed symbols.

### III. SAFE-CODIAGNOSABILITY

In this section, we present the definition of safe-codiagnosability and the “separation property” of safe-codiagnosability. As described in [11], for the purpose of diagnosis, a system with deadlocking states can be converted to a deadlock free system by adding a self-loop on a special event “$\sigma_{stop}$” with $M_i(\sigma_{stop}) = \epsilon$ at each of its deadlock state without affecting the diagnosis analysis. So without loss of generality, we assume a system to be diagnosed, also called a “plant”, to be deadlock free.

**Definition 1:** [11] Let $L$ be the prefix-closed language generated by a plant, and $K$ be a prefix-closed specifying the non-faulty plant behavior ($K = pr(K) \subseteq L$). Assume there are $m$ local sites with observation masks $M_i : \Sigma \rightarrow \Sigma_i$ ($i \in I = \{1, \cdots, m\}$). ($L, K$) is said to be codiagnosable with respect to $\{M_i\}$ if

$$\forall s \in L \cap K \exists \sigma \in \Sigma (L \cap K) \cap M_{-1} M(I) \cap L, u \in L \cap K$$

In the following lemma we provide an alternative definition of codiagnosability.

**Lemma 1:** Let $L$ and $K$ be prefix-closed plant and non-faulty specification languages respectively, and for $i \in I, M_i$ be observation mask of site $i$. Then $(L, K)$ is codiagnosable with respect to $\{M_i\}$ if and only if

$$\exists n \in \mathbb{N} : \left((L - K) \Sigma^{\geq n} \cap L \right) \cap M_{-1} M_i = \emptyset.$$

**Proof:** The condition (1) in definition of codiagnosability requires that there exist a local site $i$ such that any $st$-indistinguishable $u$ at site $i$ is faulty ($u \in L - K$). This can be rephrased as saying that it is not the case that for each site $i$ there exists a $st$-indistinguishable non-faulty trace $u_i \in K$, i.e.,

$$\neg(\forall i \in I)(\exists u_i \in M_{-1} M_i st \cap L, u_i \in K)$$

(2)

The set of traces,

$$\{w \mid \forall i \in I, \exists u_i \in M_{-1} M_i(w) \cap L, u_i \in K\}$$

is the same as the set of traces $\cap_{i \in I} M_{-1} M_i(K)$. Thus the condition (2) can be equivalently written as, $st \notin \cap_{i \in I} M_{-1} M_i(K)$. Further since $st \in L$ is a feasible extension of a faulty trace $s \in L - K$ with length of $t$ at least the delay bound $n$, $st \in L \cap (L - K) \Sigma^{\geq n}$. It follows that the definition of codiagnosability of $(L, K)$ may be rephrased as,

$$\forall n \in \mathbb{N} : \left((L - K) \Sigma^{\geq n} \cap L \right) \cap M_{-1} M_i = \emptyset.$$

**Remark 1:** We can introduce the notion of “zero delay codiagnosability” by setting $n = 0$ in the definition of codiagnosability provided by Lemma 1. Then $(L, K)$ is said to be zero-delay codiagnosable with respect to $\{M_i\}$ if

$$(L - K) \cap M_{-1} M_i(K) = \emptyset.$$  

(3)

We say a faulty sublanguage $H \subseteq L - K$ is zero-delay codiagnosable with respect to $\{M_i\}$ if $H \cap M_{-1} M_i(K) = \emptyset$.

Note that (3) is equivalent to $\cap_{i \in I} M_{-1} M_i(K) \subseteq L - K$, i.e., $(L, K)$ is zero-delay codiagnosable if and only if the non-faulty behavior $K$ is decomposable [13] with respect to the non-faulty-faulty (plant) behavior $L$.

Definition 1 captures the system property that a failure event can be diagnosed within bounded delay after its occurrence by at least one of the local sites. In order to react to a failure in a timely fashion, it is also needed that a failure be detected before system behavior becomes “unsafe”. Safe behavior includes all of non-faulty behavior and some of post-fault behavior where system performance may be degraded but still tolerable. The safety specification, denoted $K_S$, is another prefix-closed sublanguage of plant language, containing the non-fault specification, i.e., $K \subseteq K_S \subseteq L$. Then the notion of safe-codiagnosability can be formalized as follows.

**Definition 2:** Let $L$ be the prefix-closed language generated by a plant, and $K$ and $K_S$ be prefix-closed non-fault and safety specification languages contained in $L$, respectively ($K \subseteq K_S \subseteq L$). Assume there are $m$ local sites with observation masks $M_i : \Sigma \rightarrow \Sigma_i$ ($i \in I = \{1, \cdots, m\}$). $(L, K, K_S)$ is said to be safe-codiagnosable with respect to $\{M_i\}$ if

$$\forall n \in \mathbb{N} : \left((L - K) \Sigma^{\geq n} \cap L \right) \cap M_{-1} M_i = \emptyset.$$  

(4)
Just as we provided an alternative definition of codiagnosability in Lemma 1, we provide an alternative definition of safe-codiagnosability in the following lemma.

**Lemma 2:** Let $L$, $K$, and $K_S$ be prefix-closed plant language, non-fault specification, and safety specification languages respectively, and for $i \in I$, $M_i$ be observation mask of site $i$. Then $(L, K, K_S)$ is safe-codiagnosable with respect to $\{M_i\}$ if and only if

$$\exists n \in \mathbb{N} : \{(L-K)\Sigma^{\geq n} \cap L \} \cup \bigcup_{i \in I} M_i^{-1}(K) \cap K_S = \emptyset$$

**Proof:** The condition (4) in definition of codiagnosability requires that there exist a local site $i$ and a safe prefix $v \leq st$ such that any $v$-indistinguishable $u$ at site $i$ is faulty ($u \in L - K$). This can be rephrased as saying that it is not the case that for each safe prefix $v \leq st$ there exists a $v$-indistinguishable non-faulty trace $u_i \in K$, i.e.,

$$\neg(\forall i \in I)(\forall v \in pr(st) \cap K_S)(\exists u_i \in M_i^{-1}M_i(v) \cap L, u_i \in K)$$

(5) The set of traces,

$$\{w \mid \forall i \in I, \forall v \in pr(w) \cap K_S, \exists u_i \in M_i^{-1}M_i(v) \cap L, u_i \in K\}$$

is the same as the set of traces,

$$\{w \mid pr(w) \cap K_S \subseteq \bigcap_{i \in I} M_i^{-1}M_i(K), \}$$

which is the same set of traces,

$$\{w \mid pr(w) \subseteq \bigcap_{i \in I} M_i^{-1}M_i(K) \cup K_S^c\},$$

which is the set

$$\bigcup_{i \in I} M_i^{-1}M_i(K) \cup K_S^c.$$**

Note that a trace $w$ belongs to the last set if and only if $pr(w) \subseteq \bigcap_{i \in I} M_i^{-1}M_i(K) \cup K_S^c$, i.e., each prefix of $w$ has the property that it is either unsafe (belongs to $K_S^c$) or for each $i$ there exists $M_i$-indistinguishable trace $u_i \in K$.

Thus the condition (5) can be equivalently written as, $\exists n \in \mathbb{N} : \{(L-K)\Sigma^{\geq n} \cap L \} \cup \bigcup_{i \in I} M_i^{-1}M_i(K) \cap K_S = \emptyset$.

This implies, $\exists n \in \mathbb{N} : \{(L-K)\Sigma^{\geq n} \cap L \} \cup \bigcup_{i \in I} M_i^{-1}M_i(K) \cap K_S = \emptyset$.

To facilitate the development of a test for safe-codiagnosability, we show that the property of safe-codiagnosability can be separated into codiagnosability together with zero-delay codiagnosability of set of boundary safe traces, where a boundary safe trace is a safe trace which for which there exists a single-event extension that is unsafe.

**Definition 3:** Given prefix-closed plant language $L$ and safety specification language $K_S$, a safe trace $s \in K_S$ is called a boundary safe trace if there exists $\sigma \in \Sigma$ such that $\sigma s \in L - K_S$, i.e., $s \in [(L-K)\Sigma] \cap K_S$. The set of all boundary safe traces is called the boundary safe language, denoted $K_S^b$, and is given by $K_S^b = [(L-K)\Sigma] \cap K_S$.

We need the result of the following lemma.

**Lemma 3:** Consider the prefix-closed non-fault specification language $K$ and the observation masks $\{M_i\}$ ($i \in I$). Then $\bigcap_{i \in I} M_i^{-1}M_i(K)$ is prefix-closed.

**Proof:** Prefix-closure of $K$ implies prefix-closure of $M_i^{-1}M_i(K)$ for each $i \in I$. So the result follows since prefix-closure is preserved under intersection.

The following theorem presents the “separation property” of safe-codiagnosability, based on which we develop the test for safe-codiagnosability.

**Theorem 1:** Let $L$, $K$, and $K_S$ be prefix-closed plant language, non-fault specification language, and safety specification language, respectively. $(L, K, K_S)$ is safe-codiagnosable with respect to $\{M_i\}$ if and only if

1) $(L, K)$ is codiagnosable with respect to $\{M_i\}$:

$$\exists n \in \mathbb{N} : \{(L-K)\Sigma^{\geq n} \cap L \} \cap M_i^{-1}M_i(K) = \emptyset$$

2) $K_S^b$ is zero-delay codiagnosable with respect to $\{M_i\}$:

$$\bigcap_{i \in I} M_i^{-1}M_i(K) = \emptyset.$$

**Proof:** From the property of codiagnosability there exists a delay bound $n$ such that condition of codiagnosability is satisfied. We claim that the same delay bound works for the definition of safe-codiagnosability. To see this, pick $s \in L - K$ and $t \in \Sigma^*$ such that $|t| \geq n$ and $st \in L$. Then $st \in [(L-K)\Sigma^{\geq n} \cap L]$. We need to show that $st \not\in \bigcup_{i \in I} M_i^{-1}M_i(K) \cup K_S^b$, i.e., there exists a prefix $v \leq st$ such that $v \not\in \bigcap_{i \in I} M_i^{-1}M_i(K) \cup K_S^b$.

For the first case ($st \in K_S^b - K$), we can set $v = st$. Then $v$ is a prefix of $st$, and also since $v = st \in K_S$, it holds that $v \not\in K_S^b$. It remains to be shown that $v = st \not\in \bigcap_{i \in I} M_i^{-1}M_i(K)$, which holds from the property of codiagnosability since $st \in [(L-K)\Sigma^{\geq n} \cap L]$ and $[(L-K)\Sigma^{\geq n} \cap L] \cap M_i^{-1}M_i(K) = \emptyset$.

For the second case ($st \in L - K_S$), suppose for contradiction that for every prefix $v \leq st$, it holds that $v \in \bigcap_{i \in I} M_i^{-1}M_i(K) \cup K_S^b$. Since $st \in L - K_S$, there exists a prefix $w \leq st$ that is a boundary safe trace, i.e., $w \in K_S^b$. From our supposition, $w \in \bigcap_{i \in I} M_i^{-1}M_i(K) \cup K_S^b$. So,

$$w \in K_S^b \cap \bigcap_{i \in I} M_i^{-1}M_i(K) \cup K_S^b = K_S^b \cap \bigcap_{i \in I} M_i^{-1}M_i(K).$$

Then we arrive at a contradiction to the condition: $K_S^b \cap \bigcap_{i \in I} M_i^{-1}M_i(K) = \emptyset$.

$\Rightarrow$ From Lemma 2 we have,

$$\exists n \in \mathbb{N} : \{(L-K)\Sigma^{\geq n} \cap L \} \cup \bigcup_{i \in I} M_i^{-1}M_i(K) \cap K_S = \emptyset.$$

This implies, $\exists n \in \mathbb{N} : \{(L-K)\Sigma^{\geq n} \cap L \} \cup \bigcup_{i \in I} M_i^{-1}M_i(K) \cap K_S = \emptyset$.

Further from Lemma 3,

$$\bigcup_{i \in I} M_i^{-1}M_i(K) \cap K_S = \emptyset.$$**

establishing the codiagnosability.

Next to show the zero-delay codiagnosability of boundary safe traces, pick a boundary safe trace $w \in K_S^b$. Then there exists $\sigma \in \Sigma$ such that $w\sigma \in L - K_S$, and we need to show that $w \not\in \bigcap_{i \in I} M_i^{-1}M_i(K)$. Set $s = w\sigma \in L - K_S \subseteq L - K$, and pick $t$ such that $|t| \geq n$ and $st \in L$ (which is possible from our underlying assumption of plant being deadlock free). Then $st \in [(L-K)\Sigma^{\geq n} \cap L]$. From the assumption of safe-codiagnosability, $st \not\in \bigcup_{i \in I} M_i^{-1}M_i(K) \cup K_S^b$, which implies every prefix of $st$, including $w \not\in \bigcap_{i \in I} M_i^{-1}M_i(K) \cup K_S^b$.

From this it follows that $w \not\in \bigcap_{i \in I} M_i^{-1}M_i(K)$, as desired.

IV. **Verification of Safe-Codiagnosability**

From Theorem 1, we know that safe-codiagnosability can be verified by checking codiagnosability of $(L, K)$ together with zero-delay codiagnosability of $K_S^b$, the set of boundary safe traces.
Algorithm 1: Consider the finite state machine models, $G = (X, \Sigma, \alpha, z_0)$, $R = (Y, \Sigma, \beta, y_0)$, and $R^2 = (Y, \Sigma, \beta, y_0)$, respectively, of the plant, the non-fault specification, and the safety specification. The corresponding plant, non-fault specification, and safety specification languages are $L = L(G)$, $K = L(R)$, and $K_0 = L(R)$, respectively, where $K \subseteq L$. Let $M_t$ be the observation mask of site $i$ ($i \in I$). To check the safe-codiagnosability of $(L, K, K_0)$, perform the following steps:

Step 1: Check the codiagnosability of $(L, K)$ [11]
Construct a testing automaton $T = (\mathcal{G}[\mathcal{R}]) \times R \times R$ for verifying the codiagnosability of $(L, K)$. This automaton is defined as $T = (Z, \Sigma_T, \gamma, z_0)$, where

$Z = \{(x, y) \times Y \times Y\}$

$\Sigma_T = \Sigma$, where $\Sigma_T = \Sigma \cup \{\varepsilon\}$

$z_0 = ((x_0, y_0), y_0, y_0)$

$\gamma: Z \times \Sigma_T \rightarrow Z$ is defined as: $\forall \gamma = ((x, y), y_1, y_2) \in Z, \sigma_T = (\sigma, \sigma_1, \sigma_2) \in \Sigma_T - \{(\varepsilon, \varepsilon), \varepsilon, (\alpha(x, \sigma), \beta(y_1, \sigma)), \beta(y_1, \sigma_1), \beta(y_2, \sigma_2)\}$ if and only if

$[M_1(\sigma) = M_2(\sigma_1)] \wedge [M_2(\sigma) = M_2(\sigma_2)] \wedge [\alpha(\sigma, \sigma_1, \sigma_2) \in \Sigma_T]$. If the answer is yes, then $(L, K, K_0)$ is not codiagnosable, and $(L, K, K_0)$ is not safe-codiagnosable as well. Otherwise, go to the next step.

Step 2: Compute the set of “boundary safe states” $B$ in $G\parallel R$.
Construct the composition $G\parallel R$, and define the set of boundary safe states as $B = \{(x, y) \in X \times Y: \exists \sigma \in \Sigma: \alpha(x, \sigma) \neq \emptyset, \beta(\sigma, y) = \emptyset\}$. Note that if $s \in L(G) \parallel R = L(G) \cap L(R)$, then execution of $s$ results in reaching a state $(x, y) \in B$, where $x \in X$ and $y \in Y$ such that $\sigma(x) \neq \emptyset, \beta(\sigma, y) = \emptyset, s \in (L - K) \cap Y$. It follows that $s \in B$.

Step 3: Compute the zero-delay codiagnosability of $K_0^2$ with respect to $\{M_i\}$
Construct a testing automaton $T_S = (G\parallel R^2) \times R \times R$ for verifying the zero-delay codiagnosability of $K_0^2$, where $T_S$ is obtained by replacing $\mathcal{R}^2$ by the testing automaton $T$ constructed in Step 1. Let $T_S = (Z_S, \Sigma_T, \gamma_S, z_0^S)$, where $Z_S, \Sigma_S$, and $z_0^S$ of $T_S$ are defined similarly as $Z, \Sigma, \gamma$, and $z_0$ of $T$, respectively (with $\mathcal{R}$ replaced by $R^2$).

Then check if there exists an “offending state” $(x, y_1, y_2)$ in $T_S$ with $y_1 \in R$. $K_0^2$ is zero-delay codiagnosable if and only if the answer is no.

Since the correctness of the test for checking codiagnosability was established in [11, Theorem 1], in the following theorem we show the correctness of the test for checking zero-delay codiagnosability of $K_0^2$.

Theorem 2: $K_0^2$ is not zero-delay codiagnosable with respect to $\{M_i\}$ if and only if there exists a state $z = ((x, y_1), y_1, y_2)$ in the testing automaton $T_S$ with $(y_1, y_2) \in B$.

Proof: $(\Rightarrow)$ If there is a state $(x, y_1, y_2)$ in $T_S$ such that $(x, y_1) \in B$, then exist traces $s \in L(G\parallel R^2)$, $u_i \in L(R) = K$ such that $(i) s \in K_0^2 = (L - K_0) \cap Y_0 \cap K_S$, and $(ii) M_i(s) = M_i(u_i)$. This implies that $s \in K_0^2 \cap Y_0 \cap M_i(K), L_0 \cap M_i = M_i(K) \neq \emptyset$. Thus, $K_0^2$ is not zero-delay codiagnosable with respect to $\{M_i\}$.

$(\Leftarrow)$ If $K_0^2$ is not zero-delay codiagnosable with respect to $\{M_i\}$, then there exists a boundary safe trace $s \in K_0^2$ such that $s \cap M_i^{-1} M_i(K) = \emptyset$. Then execution of the trace triple $(s, u_1, u_2)$ in $T_S$ results in a state $(x, y_1, y_2)$ for each $x \in K_0^2 \cap L_0 \cap M_i = M_i(K)$.

Remark 2: Let $|X|$, $|Y|$, and $|Y_2|$ be the numbers of states in plant $G$, non-fault specification $R$, and safety specification $R_2$, respectively, and $|\Sigma|$ be the number of events. Let $L = L(G) = L(R), K_0 = L(R_S)$. Assume there are $m$ local sites. We used $|\Sigma|$ in [11] that the complexity of constructing the testing automaton $T$ and checking codiagnosability of $(L, K)$ is $O(|X| \times |Y| \times |Y_0| \times |\Sigma| + 1)$. Using a similar analysis, we can verify that the complexity for constructing the testing automaton $T_S$ and checking the zero-delay codiagnosability of $K_0^2$ is $O(|X| \times |Y_0| \times |\Sigma| + 1)$.

Remark 3: In Algorithm 1, we use two testing automata $T$ and $T_S$ to verify safe-codiagnosability of $(L, K, K_0)$. These two testing automata can be combined into a testing automaton $T' = (G\parallel R_S) \times R \times R$ by replacing $\mathcal{R}$ by $R_S \mathcal{R}$ in $T$. Then, $(L, K, K_0)$ is safe-codiagnosable if and only if there exists the “offending cycle” containing a state with the third coordinate labeled by “F”, or there exists the “offending cycle” with its first pair of coordinates contained in $B$. However, in this case, the complexity is $O(|X| \times |Y_0| \times |\Sigma| + 1)$, which is an order higher. Thus the “separation” result obtained in Theorem 1 provides an order reduction in the complexity of testing safe-codiagnosability.

Once a system is deemed safe-codiagnosable, the same methods as those presented in [11] can be applied for the synthesis of local diagnosers as well as for on-line diagnosis using them. This is because a diagnoser simply observes the plant behavior and reports a fault when it becomes certain about it. The property of safe-codiagnosability guarantees that at least one diagnoser become certain within bounded delay of the occurrence of a fault and prior to the system behavior becoming unsafe. Details are omitted here.

The following example illustrates how to verify the safe-codiagnosability using Algorithm 1.

Example 1: Figure 1 (a), (b) and (c) show a plant model $G$, a non-fault specification model $R$, and a safety specification model $R_2$, respectively. The set of events is given by $\Sigma = \{a, b, f\}$. There are two local sites, with their observation masks given as follows:

$M_1(a) = a, M_1(b) = M_1(f) = c$;
$M_2(b) = b, M_2(a) = M_2(f) = \varepsilon$.

It can be verified that $(L(G), L(R))$ is codiagnosable with respect to $\{M_i\}$ by constructing a testing automaton $T = (G\parallel R) \times R \times R$, which is omitted here.

Since $L = L(G) = pr(ab + faa)$ and $K_0 = pr(ab + fa)$, the boundary safe language $K_0^2 = ([L - K_0]) \cap Y_0 \cap K_S = \{fa\}$. Following the trace $fa$, state “3” in $G$ and state “3” in $R_2$ are reached. Thus, the set of boundary safe states is given by, $B_1 = \{(3, 3)\}$. Figure 1 (d) shows a part of the testing automaton $T_S = (G\parallel R_2) \times R \times R$, where an offending state “3” in $R_2$ is reached. Therefore, $K_0^2$ is not zero-delay codiagnosable with respect to $\{M_i\}$, and thus $(L, K, K_0)$ is not safe-codiagnosable with respect to $\{M_i\}$ as well.

Now, if we relax the safety requirement by considering a new enlarged safety specification model $R_2$, as shown in Figure 2 (a), the system becomes safe-codiagnosable. To see this, $K_0 = pr(ab + f + fa)$, the boundary safe language is given by, $K_0^2 = ([L - K_0]) \cap Y_0 \cap K_0 = \{fa\}$. Thus, the set of boundary safe states is given by, $B_2 = \{(4, 4)\}$. The new testing automaton $T_2 = \{(4, 4)\}$. The new testing automaton $T_2 = \{(4, 4)\}$.
Future work will consider extension applying the inferencing based approach [7], [16].

REFERENCES